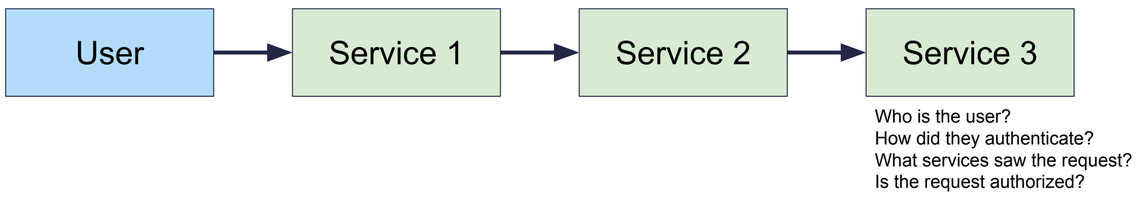
Workload Identity in Multi System Environments

IETF 118 – Tuesday, 13:00-15:00 – Congress Hall 1

Introduction

Modern large-scale systems are typically composed of many “microservices,” or small software components that interact via APIs and collectively compose a user-facing system. While the microservices approach can improve scalability, availability, and maintainability in these systems, it can present unique challenges for security because identifying the original source and intermediate steps of a request (after multiple hops) may be impossible.



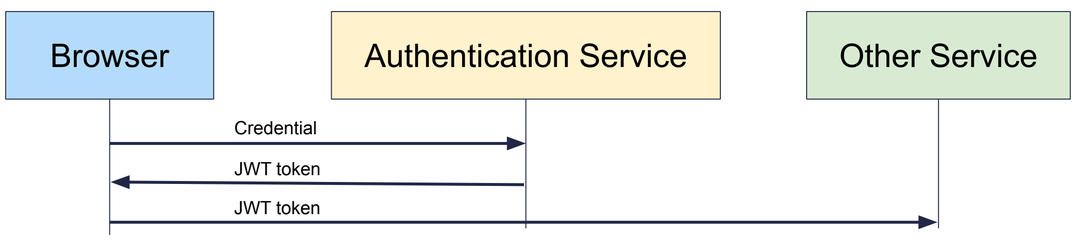
The **Workload Identity in Multi System Environments** BOF aims to address the challenges associated with implementing fine-grained, least privilege access control for workloads deployed across multiple service platforms, spanning both public and private clouds using existing standards, open source projects, and community practices.

Problem Statements

- How can we **attest** the identity of **software services** involved in processing a request?
- How can we **attest** the identity of the **hardware/cloud environment** involved in processing a request?
- How can we **integrate** with existing user authentication systems?
- How can we **authorize** a request based on the user and the chain of services that the request went through?
- How can we do these things using **existing technologies** as much as possible?
- How can we make **widely applicable best practices** to solve these problems?
- Can we **improve** existing technologies to make them work together better?

Existing Standard: JWT

JSON Web Token (JWT, RFC 7519) is an open standard for securely transmitting signed JSON data between parties. JWTs can either be signed using symmetric encryption (HMAC) or asymmetric encryption (RSA or ECDSA). [1]

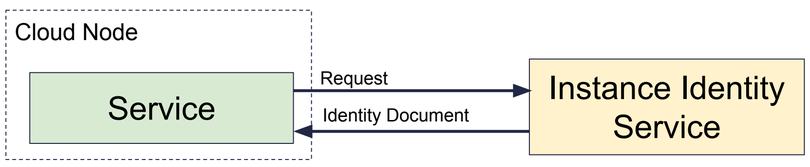


JWTs are currently widely used as a part of the user authentication flow for web services. JWTs are bearer tokens and are vulnerable to replay attacks.

Existing Technology: Cloud Instance Identities

Each major cloud vendor provides some mechanism for software running on a node to obtain the identity of a node. [2, 3, 4]

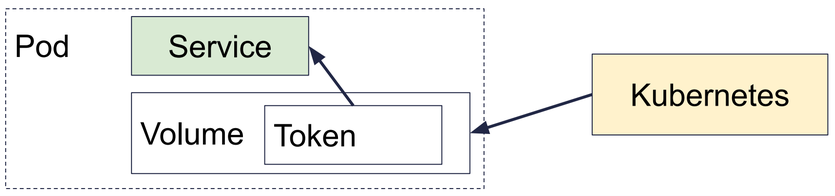
Initially, these consisted of simple JSON documents containing signed instance metadata. This is reachable through an instance metadata endpoint that has minimal security. Each cloud vendor uses a different endpoint and provides a different metadata format.



As cloud providers have matured and expanded their offerings, some now offer more secure mechanisms to provide an X.509 certificate to workloads in a secure manner. The details of how they are assigned vary greatly. [5]

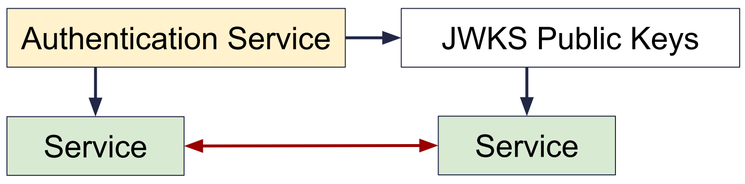
Existing Technology: Kubernetes Service Account Tokens

Kubernetes is a widely used runtime environment for cloud services. It provides a mechanism called “service account tokens” which allows any service to obtain a bearer token from a special volume [6].



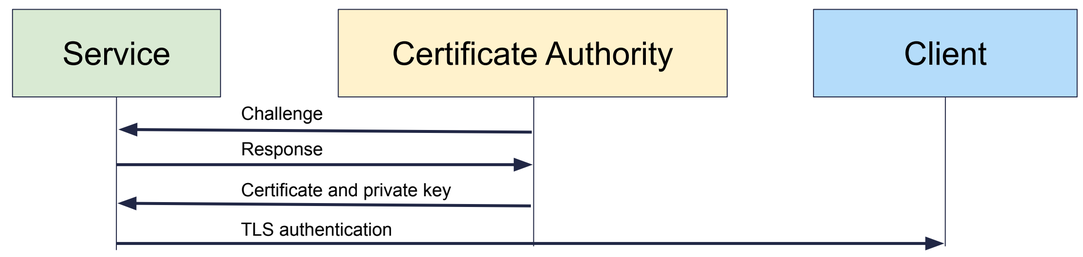
Existing Standard: OpenID Connect

OpenID Connect (OIDC) is a standard for building distributed authentication systems using JWTs built on top of OAuth (RFC 6749). OIDC is lightweight, flexible, and often used for service authentication. It also allows federation between multiple authentication services. [7, 8]



Existing Standard: ACME

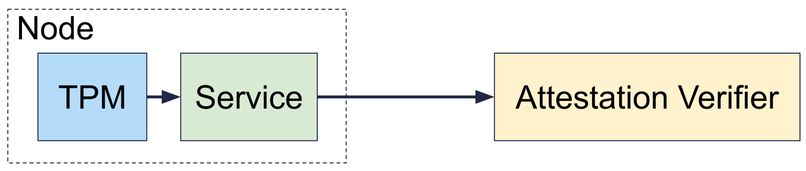
Automated Certificate Management Environment (ACME, RFC 8555) is a widely used standard for automatically distributing X.509 certificates. A certificate authority provides a challenge; a server answers the challenge and receives a certificate and private key. [9]



ACME is widely used for public-facing services. One gap for multi system environments is that It does not attempt to distinguish multiple services running on the same node or network address.

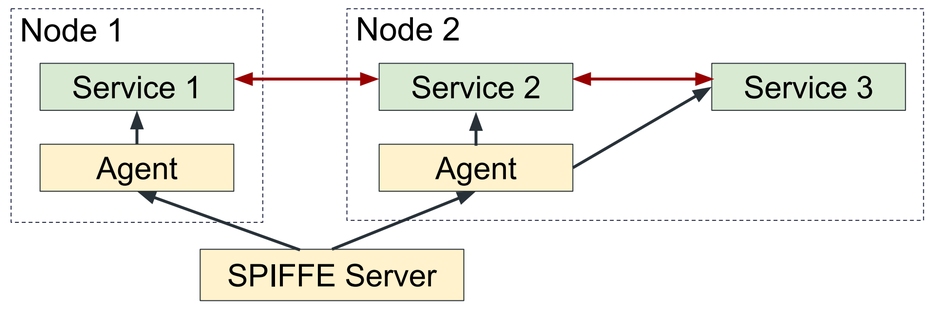
Existing Standard: Trusted Platform Module 2.0

Most modern computer hardware includes a Trusted Platform Module which provides an array of cryptographic services. This includes an Endorsement Key, LDevID, and IDevID which can be used to identify to a third party. Recently, cloud vendors and virtual machine manager vendors have started to support TPM for individual virtual machines using “virtual TPMs.” [10, 11].



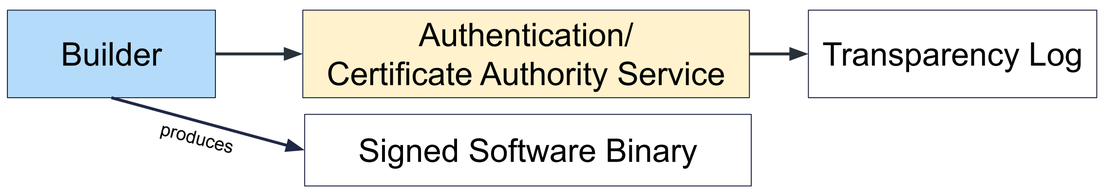
Emerging Standard: SPIFFE

The Secure Production Identity Framework for Everyone (SPIFFE) is a standard way to provide unique credentials (tokens or X.509 certificates) to services [12]. SPIFFE works by first securely identifying the node that the service is running on; then securely identifying the service itself; then providing unique credentials (both JWTs and X.509 certificates) to that service.



Emerging Standard: sigstore

Sigstore is a standard for software signing in multi-service environments. The key idea in Sigstore is that components are signed at build time using an ephemeral certificate and private key, which is then stored in a Certificate Transparency log (RFC 6962). [13]



Emerging Standard: Tokens with Attestations

Multiple newer efforts are emerging to create token types to replace JWT that have more flexible sets of attestations for complex multi-service environments.

- Macaroons are a standard token time that contains attestations that can be attenuated [14]
- Biscuits are an extension of Macaroons that use a logic programming language (Datalog) to include sophisticated, flexible authorization logic. [15]
- Transaction Tokens are an extension of the existing OAuth authorization framework that adds a traceable “authentication context” to all requests that trigger from a single external OAuth request. [16]

Conclusion

Modern service environments consist of large numbers of microservices, distributed across many different nodes. This presents a challenge for security because it is difficult to determine the true origin or pathway of any request. Multiple existing and new technologies attempt to solve parts of this problem.

The WIMSE BOF exists to identify best practices in implementing multi-service security using these technologies, and propose improvements when necessary.

References

1. JSON Web Token (JWT). RFC 7519
2. Amazon Web Services Instance Identity Document
3. Google Cloud Platform Compute Metadata
4. Azure Instance Metadata Service
5. Amazon Web Services IAM Roles Anywhere
6. Kubernetes Service Account Tokens
7. OpenID Connect Specification
8. OAuth. RFC 6749
9. ACME. RFC 8555
10. TPM 2.0 Standard
11. 802.1 AR Standard
12. SPIFFE Standard
13. Sigstore
14. Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud
15. Biscuit token specification
16. Transaction Tokens